Friendly Node Numbers

Addition to LOOPREQM

Thu, Mar 13, 2003

The LOOPREQM local application monitors ongoing request activity, building a list of the number of requests that are active from each requesting node. Using the PAGEPMEM page application, one can easily list out the contents of this table. But the node numbers shown in the table are pseudo node numbers, which are not easily understood, since they depend upon the position occupied by that node in the local node's IPARP table at that time. This note describes a change in LOOPREQM to provide a more easily interpretable table.

The table already maintained by the LA is needed for its own monitoring purposes, since it watches the arrival of received datagrams as recorded in the NETFRAME data stream, and those data stream records have a node number field that is a pseudo node number.

The scheme suggested here is to build another table that contains the information about active requests by node, but format it in another way that is easier to read; *i.e.*, leave the old table in place, but build a new table that is easier to read.

Inspired by the code in the PAGENETF application that produces a listing showing the recent NETFRAMES activity, the scheme tries to translate the pseudo node number into a native node number, or an Acnet node number, in cases where there is no native node number. The means of doing this is supported by a special listype (95) that takes a pseudo node number ident and returns up to 16 bytes of information about that pseudo node number. The client making such a request then builds a replacement node number as appropriate. The PAGENETF application uses the notation 0xCnnn for Classic port, or 0xAnnn, for the Acnet port, where nnn is a 3-digit hexadecimal node number that is a native node number, or in the case that none is available, an Acnet node number. All of the front ends running this software have both native node numbers in the range 0x0500–0x07EF, and Acnet node numbers in the range 0x0900–0CEF, whereas an Acnet console client node has only an Acnet node number. A PC or Macintosh client has neither, so it can only be represented via the original pseudo node number.

After the table of active nodes has been built, once per second, the new secondary table can be built by merely building a request using listype 95 for each of the pseudo node numbers in the table of active nodes. The reply data can be obtained immediately after making the request, since it is by definition a local request. Then the new table can be built.

The format of the new table can be simplified from the current table. The only fields needed are the friendly node number, the count of active requests, and the request block type number of the first one. The latter two fields can be squeezed into two bytes, so that the new table entries need only be 4 bytes in size. If desired, the new table could be sorted by node number. The LOOPREQM local application currently assumes a maximum number of requesting nodes of 32. The new table, then, could fit into only 128 bytes.

Considered an alternative new format. Suppose it were of interest to keep an active list of all requests known to the system, more than merely the number of active requests from each requesting node. During the scan of active requests made by the LOOPREQM logic, a new table could be constructed with more information included. For the RETDAT protocol, the three-word request header would be interesting, as it includes the number of bytes to be delivered in replies, the number of Acnet devices, and the Frequency-Time Descriptor, or FTD. For

Classic requests, one could try to provide analgous information. The following might be a suitable format for an 8-byte record:

Field	*Size*	*Meaning*
`fNode`	2	friendly node number
`nBytes`	2	#bytes returned in replies
`nDev`	2	#devices in request
`ftd`	2	copy of `FTD`, or equivalent

Some of the records might be for Classic protocol, and others for Acnet protocol. The friendly node number should ordinarily indicate which one in its most significant 4 bits. For the case of a PC that can speak both protocols, but has no friendly node number, it may more more difficult. From experience, however, this is usually not a problem.

The new list would be updated at 1 Hz, so it cannot show fleeting request activity, especially one-shot requests. But one can capture such `RETDAT` requests by examining the `RETDAT` request log in data stream named “`RETDLOG  `“. The system `RETDAT` support logs a record into this data stream every time a `RETDAT` request is received and accepted.